Computer Science Tripos – Part II – Project Proposal

Comparing Machine Learning Techniques for Mobility Graph Classification

Miteyan Patel, Robinson College
Originator: Dr Sandra Servia
20th October 2017

Project Supervisor: Dr Sandra Servia
Director of Studies: Prof Alan Mycroft
Project Overseers: Dr Ian Wassell & Dr Rafal Mantiuk

Introduction

The pervasiveness of smartphones has generated large amounts of data, whose analysis can disclose valuable information regarding users' behaviour. At the same time, recent advances in machine learning, especially on generalising the use of Convolutional Neural Networks (CNN) on graphs, have allowed extending the use of deep learning beyond the NLP and image recognition domains. In this project, we will explore the performance of different machine learning techniques to the task of inferring demographics (age, gender, job type) of the individuals from a graph representation of their mobile GPS location data. The best performing model will be trained on the server and ported to a mobile phone application which collects location data, builds mobility graphs from it and uses the model and graphs to infer the demographics of the user locally on their device.

The main motivation of the project is to select a machine learning model and provide a mobile framework for classifying users according to their mobility patterns, whilst keeping their location data on their devices in order to prevent privacy leaks. A similar approach has applicability to identifying stages of Alzheimer's Disease given a suitable data set, since it has been observed that getting lost and misremembering the location of places are symptoms that typify the onset of the disease, therefore their mobility graphs are likely to differ from the neurotypical which could be used for early diagnoses of the disease before symptoms such as dementia and irreversible brain damage occur.

Two main issues need to be addressed to build such a system for this project. First, the selection of a supervised classifier to infer the user's demographic label. Recent advances in Machine Learning have generalised CNNs from regular grid-like data such as images and speech to irregular data like graphs as inputs [1, 2, 3]. For this project, we will explore the use of graph CNNs to infer individuals' demographic label from graph representations of their mobile GPS location data collected with their mobile devices. We will study how graph CNNs compares to traditional supervised learning approaches, from Logistic Regression to Support Vector Machines, based on feature extraction of the graphs. We will evaluate the performance by comparing the memory required, the speed of training and making predictions with the models, and accuracy of the different approaches after

optimising each. For the input data into the models, we will create graphs with nodes for locations and edges as transitions between locations in the period of observation.

Second, we need to provide a privacy-aware solution that guarantees the confidentiality of the individuals sensitive location data and prevent privacy leaks by inferring the label locally on the mobile device. To do so, we will build and train the model on a server by using the dataset. This model will be then ported to a mobile phone application that collects the location data, builds the graph and infers the demographic label locally. We will also evaluate the performance of the model on the mobile application regarding its time to build location graphs and infer the users class, as well as the battery consumption.

Starting point

I have previous knowledge about machine learning through the **Artificial Intelligence** course and reading in my spare time, although for a deeper understanding of the machine learning algorithms and knowledge about making optimizations for increasing the performance of the algorithms I aim to use the Machine Learning by Stanford online course[4]. There is also a **Machine Learning and Bayesian Inference** course in Lent. Although I intend to complete most of the work concerning these algorithms before the start of the course and learn about other additional algorithms from other sources. I have previous knowledge of Python, TensorFlow, Scikit-learn [8, 9] through small projects and reading, although I aim to improve these skills through the online resources such as the Deep Learning courses on Udacity and Coursera[5, 6]. I also intend to read relevant papers, blog posts, videos to further understanding. **Algorithms**, **Software Engineering**, **OOP**, **Mathematical Methods for Computer Science** are courses that will be useful throughout the project.

I will be using the Nokia: The Mobile Data Challenge: Big Data for Mobile Computing Research dataset[7] containing location in latitude longitude collected from smartphone GPS sensors in Lausanne, Switzerland, for over a year. It contains demographic labels such as gender, age group, marital status, job type, and number of people in the household for some users.

I will be investigating the potential of a recent paper: Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering [1]. It contains an implementation which I will be modifying for use on the mobility graphs constructed because it would be difficult to implement a better version from scratch and therefore will give me better results, whilst allowing me more time to try out other possible traditional algorithms and determine if we need the advanced graph CNN algorithm to accurately predict a users class or if we could just use feature extraction and standard supervised classifiers for a similar testing error and compare the trade-offs. The implementation will be used under the MIT license and the paper will be cited[1].

I will use an existing script to extract graphs from raw GPS data from the Nokia dataset to start with. This will allow me to earlier address the main aims of the project, leaving the investigation of how to build better trajectory graphs from GPS traces as a possible

extension for this project. There is an application for Android that collects GPS location coordinates that will be greatly extended to build graphs, extract possible features and infer the users class.

Libraries

- Android SDK: Development environment for Android. I am already familiar with this from previous projects and the group project.

- TensorFlow [8]: Open source deep learning library, to assist in the implementation of the algorithms and used in the graph CNN paper. TensorFlow is widely used in industry and research as the CNN paper uses TensorFlow [1] so my understanding of TensorFlow is paramount to understand and modification of the open source implementation, it also allows a model to be ported to Android.

- Scikit-learn [9]: Another machine learning library. Will allow experimenting with more algorithms and are optimized for faster training and would work better than an implementation from scratch, also contains API for porting to Android. This will help meet my success criterias earlier and possibly allow the extensions to be implemented.

Resources required

The project will be carried out on my personal laptop, a mid 2014 MacBook Pro running macOS Sierra. However, the computation will be done in the server *pizza.cl.cam.ac.uk* that I will be given access to for the use of the restricted dataset[7]. I also have my own Android Nexus 5X smart phone which I can use to test and build the application. If it breaks I will get a new Android phone.
The dataset is available only for use on this server, I have permission to access the server and dataset from my supervisor and through the system administrators at the Computer Lab. The University has an agreement in place to use the data until December 2018. I have signed a Confidentiality Disclosure Agreement with Idiap Research Institute for the use of the dataset and adhere to the strict rules about the restricted data and its privacy. I will have get approval for the ethical review of research with human participants for the use of personal information from the dataset and also for potential tests of the application for Android if time permits.

I will use a GitHub repository, with separate branches for the mainline, features and bug fixed for organising the project and to commit code only a few hours after completion to prevent losing more than a few hours of code in case my machine breaks. It also allows the ability to track bugs, control versions, and host the code in a safe, easily accessible place in case my machine breaks. I will also make backups every major update onto several USB sticks.

Work to be done

The project breaks down into the following sub-projects:

1. Understanding the graph Convolutional Neural Network algorithm proposed in [1] and modifying the open source implementation accordingly, as well as the input pipeline to classify the location graphs. These graphs will be previously extracted from the Mobile Challenge Dataset location data [7] and divided into training, validation and test set.

2. Identifying relevant features from the mobility graphs that will be used for inputs into the supervised learning algorithms. These will be different measurements of centrality, shortest path, centrality, etc.

3. Implementing machine learning algorithms in Python using TensorFlow and scikit-learn[8, 9]. These will use, as input data, the relevant feature identified in 2. These may include the following but are not limited to and likely to change: Logistic Regression, Support Vector Machines, Naive Bayes classifier, Neural Networks, Decision Trees, Random Forests.

4. Optimising the different trained models for accuracy, time and space. Identifying the best performing model to be ported to the mobile phone application.

5. Modifying the implementation for use in the Android mobile phone application using the TensorFlow Android API and completing the Android app to construct graphs locally from the location data and use the model to locally infer the label. Evaluate the time it takes for the building of the graph from location data, the time it takes to make the class inference for the user, as well as the battery consumption.

Success criteria

At the end of the project I should have completed the following criteria to allow the project to be considered successful:

1. Applied, optimized and tested the graph Convolutional Neural Network algorithm to the location graphs constructed from raw location data to classify the user into their demographic class[1].

2. Trained, optimized, and benchmarked the performance of the supervised machine learning algorithms on features extracted from mobility graphs.

3. Evaluated the models quantitatively and chosen the most appropriate of these and pre-train the model and port into the Android application.

4. Have an Android application which can construct mobility graphs from the users location and infer the demographic class of the user locally using the model.

Possible extensions

If I achieve my main result early, I will explore how to construct different types of graphs from the raw location data. The nodes can be represented as either latitude and longitude pair or an area of the coordinates. The edges could include the direction of the user or be undirected. We could also add weights to edges based on popularity of the location, or omit less popular or more popular locations. Other possibility would be to create synthetic data for Alzheimer's patients based on differences in movements between people with memory impairing diseases such as Alzheimers disease and the neurotypical, by sensing changes in their tracking graph for early diagnosis. I might also use the graph2vec [11] algorithm to convert the graphs created into a vector which can be directly used by the traditional machine learning algorithms, or try other graph CNN algorithms [2, 3], although their open source implementation has not been released yet.

Further extensions could include creating a distributed machine learning model[12], where the model would be trained cooperatively and successive updates of the model would happen locally on the users' devices. This would ensure the confidentiality of the location data since it is not sent to a server, but only the weights of the trained model are disclosed. However, this would involve many changes and extensions to the existing system and therefore it may be more appropriate as future work after the dissertation time frame.

Timetable

- **Michaelmas weeks 0-2 — October 6th - 20th**
  - Refine the project idea and write project proposal drafts.
  - Make changes to draft and project based on feedback.
  Milestones: Have a finished Project Proposal submitted by October 20th.

- **Michaelmas weeks 2-4 — October 21st - November 3rd**
  - Background reading on TensorFlow, Scikit-learn, supervised machine learning techniques, Naive Bayes classifier, SVM, Neural Networks, Decision Trees, graph feature extractions, graph CNN algorithms
  - Set up GitHub repository, back up storage solutions for the project in case of failures.
  Milestones: Studied the online courses[4, 5, 6] and background reading needed for the theory and skills to complete the project.

- **Michaelmas weeks 4-6 — November 4th - 17th**
  - Construct graphs from raw location data.
  - Split into training and test sets.
  - Create a dissertation layout LATEX document.
  - Read the graph CNN paper[1].
  Milestones: Have working training, test, validation datasets for the graph CNN algorithm. Understood the graph CNN algorithm.

Michaelmas weeks 6-8— November 18th - December 1st
- Modify the Graph CNN algorithm for the location graph dataset.
Milestones: Started working on training model for the graph CNN.

- **Christmas vacation weeks 0-2 — December 2nd - 15th**
  - Continue work on the graph CNN and train, optimize, benchmark, evaluate.
  - Extract Features from graphs for traditional supervised learning tasks and make test, train, validation dataset.
  - Start writing up the Introduction chapter.
  Milestones: Finished and have a working trained model for the graph CNN and so have met the first success criteria. Have working training, test, validation datasets to allow development of machine learning algorithms.

- **Christmas vacation weeks 2-4 — December 16th - 29th**
  - Implement, train, optimise, test Naive Bayes, Logistic Regression and Neural Network classifiers.
  - Start writing Preparation chapter.
  Milestones: Have working Naive Bayes, Logistic Regression and Neural Network models with metrics for training data to allow for evaluation later on. Finished writing the introduction chapter draft.

- **Christmas vacation weeks 4-6 — December 30th - January 12th**
  - Implement, train, optimise, test SVM, Decision Tree, Random Forest classifiers, or other through research.
  Milestones: Have working SVM, Decision Tree, Random Forest models for training data with metrics to allow for evaluation later on. Finished the preparation chapter.

- **Lent weeks 0-2 — January 13th - 26th**
  - Evaluate and compare all of the machine learning models.
  - Start creating application for Android to construct graph from location data then infer the class of the user.
  - Started writing the implementation chapter.
  Milestones: Evaluated and chosen the best machine learning model to be exported to the Android application and so have met the second success criteria. Finished the preparation chapter draft.

- **Lent weeks 2-4 — January 27th - February 9th**
  - Start, complete and send the progress report by February 2nd and be ready for the overseers presentation.
  - Make and practice the overseers presentation.
  - Train and export machine learning model to application to make predictions locally.
  Milestones: Sent the progress report and have practiced the presentation to overseers. Exported the best model to Android, using TensorFlow Android API or scikit-learn open source API [8, 10], to port pre-trained model and met the third success criteria.

- **Lent weeks 4-6 — February 10th - February 23rd**
  - Complete, test and debug the application for Android.
  - Start writing the evaluation chapter.
  Milestones: Tested the application to make predictions the application for Android

and met the last success criteria. Finished writing the implementation chapter.

- **Lent weeks 6-8— February 24th - March 9th**
  - Finish up any existing implementation work and extensions.
  - Start writing conclusion.
  Milestones: Finished up any implementation and extension work for the dissertation. Finished evaluation chapter.
- **Easter vacation weeks 0-2 — March 10th - 23rd**
  - Buffer slot for delays, testing and work due and potentially for implementing extensions if time permits.
  - Finish dissertation draft and send to my supervisor and DoS for initial feedback.
  Milestones: In case of delays not planned for encountered during the project. Submitted a dissertation draft for initial review.
- **Easter vacation weeks 2-4 — March 24th - April 6th**
  - Make changes for the dissertation from comments.
  Milestones: Updated dissertation based on comments from DoS and supervisor.
- **Easter vacation weeks 4-6 — April 7th - 20th**
  - Buffer slot in case of delays.
  - Finish writing dissertation.
  Milestones: Finished dissertation and ready for hand-in.
- **Easter term weeks 0-2 — April 21st - May 4th**
  - Buffer slot in case of delays.
  - Submit dissertation.
  Milestones: Handed in the dissertation by May 4th for the deadline on May 18th.

References

[1] Defferrard, M., Bresson, X.,& Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in Neural Information Processing Systems (pp. 3844-3852).

[2] Niepert, M., Ahmed, M., & Kutzkov, K. (2016, June). Learning convolutional neural networks for graphs. In International Conference on Machine Learning (pp. 2014-2023).

[3] Levie, R., Monti, F., Bresson, X., & Bronstein, M. M. (2017). CayleyNets: Graph Convolutional Neural Networks with Complex Rational Spectral Filters. arXiv preprint arXiv:1705.07664.

[4] Stanford University: Machine Learning course
`https://www.coursera.org/learn/machine-learning`

[5] Deep Learning Python TensorFlow course
`https://www.udacity.com/course/deep-learning--ud730`

[6] Deep Learning Python sklearn course
`https://www.coursera.org/learn/neural-networks-deep-learning/`

[7] Kiukkonen, N., Blom, J., Dousse, O., Gatica-Perez, D., & Laurila, J. (2010). Towards rich mobile phone datasets: Lausanne data collection campaign. Proc. ICPS, Berlin.

[8] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., & Ghemawat, S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

[9] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake Vanderplas, Arnaud Joly, Brian Holt, Gal Varoquaux. API design for machine learning software: experiences from the scikit-learn project. arXiv:1309.0238.

[10] Scikit-learn porter for Java

`https://github.com/nok/sklearn-porter`

[11] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, Shantanu Jaiswal (Jul 2017). graph2vec: Learning Distributed Representations of Graphs. arXiv:1707.05005

[12] Shokri, R., & Shmatikov, V. (2015, October). Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (pp. 1310-1321). ACM.